#############################################################################

#

# Logging settings

#

set logGroup "ConfigurationManager"

set logLevelWarning warning

set logLevelInfo info

set logLevelDebug debug

log $logGroup $logLevelInfo "Parsing configuration script"

#############################################################################

#

# Constants

#

seti Off 0

seti On 1

seti Low 1

seti Medium 2

seti High 3

seti MediumButDefaultLow 4

seti memoryLevelHigh 512

seti memoryLevelMedium 384

seti memoryLevelLow 256

seti memoryLevelUnknown 0

if ( match("${osVersion}","\*NT 6.0\*") )

 seti cpuLevelHigh 3000

 seti cpuLevelMedium 2600

 seti cpuLevelLow 2000

 log $logGroup $logLevelInfo "Setting cpu level for Vista."

else

 seti cpuLevelHigh 2800

 seti cpuLevelMedium 2200

 seti cpuLevelLow 1450

 log $logGroup $logLevelInfo "Setting cpu level for non-Vista."

endif

seti cpuLevelUnsupported 0

#############################################################################

#

# Identify gfx device

#

# set some config variables based on a card table and vendor specific rules

# sets isCardFound, cardVendor, and cardName

include "Video Cards.sgr"

# Fallback on the card name text supplied by the card itself.

# Assumes that at least the cardVendor has been matched, since

# vendor name is not contained in some vendor card names.

# Note that specific vendors are overridden to a default card.

# For ATI -> Radeon 9800 Pro

# For NVIDIA -> NVIDIA 4800 TI SE

if (not $isCardFound)

 set cardName $cardNameText

endif

#############################################################################

#

# Configuration override controlled by command line option -cardConfig.

#

if (varExists(cardConfig))

 # test unsupported cards using sw rendering

 if ($cardConfig = 1)

 log $logGroup $logLevelInfo "Testing with all devices unsupported."

 set cardVendor TestingAllUnsupported

 set cardName TestingAllUnsupported

 elseif (($cardConfig = 2) and ($deviceNumber = 0))

 log $logGroup $logLevelInfo "Testing with the first device unsupported."

 set cardVendor TestingFirstUnsupported

 set cardName TestingFirstUnsupported

 elseif (($cardConfig = 3) and ($deviceNumber != 0))

 log $logGroup $logLevelInfo "Testing with the all devices except first unsupported."

 set cardVendor TestingAllButFirstUnsupported

 set cardName TestingAllButFirstUnsupported

 elseif (($cardConfig = 100) and ($deviceNumber = 0))

 log $logGroup $logLevelInfo "Testing first device as swvp only part."

 boolProp forceSoftwareVP true

 endif

endif

#############################################################################

#

# Determine if gfx device is supported, force software rendering if not

#

setb useSoftwareRasterizer false

if (match("${cardVendor}", "ATI"))

 if (match("${cardName}", "\*Mach\*") or match("${cardName}", "\*Rage\*"))

 setb useSoftwareRasterizer true

 endif

elseif (match("${cardVendor}", "NVidia"))

 if (match("${cardName}", "NV 1") or match("${cardName}", "NV 2") or match("${cardName}", "\*Riva\*") or match("${cardName}", "\*TNT\*"))

 setb useSoftwareRasterizer true

 endif

elseif (match("${cardVendor}", "Intel"))

 if (match("${cardName}", "\*810\*") or match("${cardName}", "\*815\*") or match("${cardName}", "\*740\*") or match("${cardName}", "\*752\*"))

 setb useSoftwareRasterizer true

 endif

elseif (match("${cardVendor}", "S3"))

 if (not match("${cardName}", "\*GammaChrome\*") and not match("${cardName}", "\*DeltaChrome\*"))

 setb useSoftwareRasterizer true

 endif

else

 # unsupported/unknown vendor

 setb useSoftwareRasterizer true

endif

if ($useSoftwareRasterizer)

 log $logGroup $logLevelWarning "Unsupported video card. Forcing software rendering on this device"

else

 log $logGroup $logLevelInfo "Supported video card."

endif

if (not $useSoftwareRasterizer)

 # failed to obtain device texture memory size, force to 32MB

 if ($textureMemory = 0)

 seti textureMemory 32

 setb textureMemorySizeOK false

 endif

 if ($textureMemory < 28)

 log $logGroup $logLevelWarning "Insufficient video memory. Forcing software rendering on this device"

 # require a card with at least 32MB

 setb useSoftwareRasterizer true

 endif

endif

if ($useSoftwareRasterizer)

 # ignore texture memory reported by gfx device, this is not hardware texture memory

 seti textureMemory 32

endif

if (not $useSoftwareRasterizer)

 log $logGroup $logLevelInfo "Hardware rendering is enabled"

endif

#############################################################################

#

# Apply gfx device specific workarounds

#

# Available device props:

# forceSoftwareVP

# forceSoftwareDevice

# queriesOK

# windowedModeFormatConversionOK

# enumerateMultisampleLevels

# skipValidateDevice

# enableDriverMemoryManager

# softwareDeviceLocalVideoMemorySize

# deviceLocalVideoMemorySizeDefault

# disableVSyncSupport

# substituteFlipForDiscardSwapEffect

# minTextureDimensionForBalance

#

# Additional props:

# vs2LoopsFunctional

setb textureMemorySizeOK true

setb supportsDirtyRect true

setb supportsTurboRect true

setb supportsSpecialEventCamera true

setb forceLowSettings false

setb forceLowResolution false

setb defaultLowResolution false

setb forceMediumMaterialDetail false

setb isIntegratedChipset false

setb forceMediumShadows false

setb forceMediumObjectDetail false

boolProp useRenderTextures false

uintProp antialiasingSupport 1

boolProp dontMergeNHFlora true

if (not $useSoftwareRasterizer)

 # never trust the driver to manage its own memory

 boolProp enableDriverMemoryManager false

 boolProp vs2LoopsFunctional false

 boolProp presentWorkaround false

 boolProp enumerateMultisampleLevels true

 # not enough texture memory for antialiasing

 if ($textureMemory < 127)

 boolProp enumerateMultisampleLevels false

 endif

####################### NVIDIA ######################################################################

 if (match("${cardVendor}", "NVIDIA"))

 # on NVidia cards, create a dummy texture on device creation to prevent BSODs

 boolProp createNVidiaWorkaroundTexture true

 if ($appControlledAA)

 # nvidia drivers handle offscreen aa

 uintProp antialiasingSupport 3

 else

 # remove this when the texture manager balancing is fixed.

 boolProp disableTexMemEstimateAdjustment true

 # turn off dirty rects

 uintProp antialiasingSupport 1

 setb supportsDirtyRect false

 endif

 if (match("${cardName}", "\*GeForce2\*Integrated\*") or match("${cardName}", "\*GeForce?2\*Integrated\*") or match("${cardName}", "\*GeForce4\*Integrated\*") or match("${cardName}", "\*GeForce?4\*Integrated\*") or match("${cardName}", "\*GeForce2 Quadro2 Pro\*"))

 boolProp causticsEnabled false

 boolProp enumerateMultisampleLevels false

 setb forceLowSettings true

 setb forceLowResolution true

 endif

 # the 6200 Turbo Cache peformed very poorly in CATLAB runs for EP2

 if (match("${cardName}", "\*6200 TC\*") or match("${cardName}", "\*6100\*"))

 setb forceLowSettings true

 setb defaultLowResolution true

 endif

 # need to get around the problem of z fighting in GeForce4 cards

 if (match("${cardName}", "\*GeForce4\*"))

 boolProp raiseLightSnowFF true

 endif

 # disable shaders by default for NV3x generation, only enable for high end cards

 if ($maxVertexProgramVersionHWMajor = 2)

 boolProp useShaders false

 if (match("${cardName}", "GeForceFX\*") or match("${cardName}", "\*GeForce?FX\*"))

 if (match("${cardName}", "\*57?0\*") or match("${cardName}", "\*58?0\*") or match("${cardName}", "\*59?0\*"))

 boolProp useShaders true

 boolProp causticsEnabled false

 log $logGroup $logLevelInfo "Enabling shaders for high-end NV3x"

 endif

 if (match("${cardName}", "\*NV35\*") or match("${cardName}", "\*NV36\*") or match("${cardName}", "\*NV38\*") or match("${cardName}", "\*NV39\*"))

 boolProp useShaders true

 log $logGroup $logLevelInfo "Enabling shaders for high-end NV3x"

 endif

 # 5700 cards sometimes have problems with bump mapping so setting default to medium

 if (match("${cardName}", "\*5700\*"))

 log $logGroup $logLevelInfo "Forcing Medium Material detail on 5700"

 setb forceMediumMaterialDetail true

 endif

 if (match("${cardName}", "\*5700LE\*"))

 log $logGroup $logLevelInfo "Forcing low settings on 5700LE and 6100"

 setb forceLowSettings true

 boolProp useShaders false

 endif

 endif

 if (match("${cardName}", "QuadroFX\*") or match("${cardName}", "\*Quadro?FX\*"))

 if (match("${cardName}", "\*3000\*") or match("${cardName}", "\*1?00\*"))

 boolProp useShaders true

 log $logGroup $logLevelInfo "Enabling shaders for high-end NV3x Quadro"

 endif

 if (match("${cardName}", "\*2000\*") or match("${cardName}", "\*1000\*") or match("${cardName}", "\*1300\*"))

 boolProp causticsEnabled false

 log $logGroup $logLevelInfo "Disabling caustics as these cause cards cause problems with shadows"

 endif

 endif

 # caps read vs 3.0, but dx9b is returning vs 2.0, once 9c is used this fix can be removed

 if (match("${cardName}", "\*GeForce\*"))

 if (match("${cardName}", "\*68?0\*"))

 boolProp useShaders true

 log $logGroup $logLevelInfo "Enabling shaders for high-end NV40 Parts on Dx9b"

 endif

 endif

 endif

 if ($maxVertexProgramVersionHWMajor = 0)

 boolProp skipValidateDevice true

 log $logGroup $logLevelInfo "Enabling D3DERR\_CONFLICTINGRENDERSTATE validation workaround"

 endif

####################### RADEON ######################################################################

 elseif (match("${cardVendor}", "ATI"))

 # loops support broken in 6458. Fixed in catalyst 4.9 (6476) but we require 4.10 to be safe.

 if ($driverBuild >= 6476)

 boolProp vs2LoopsFunctional true

 endif

 # workaround for forced aa crash as of 6458. Fixed in catalyst 4.9 (6476) but we require 4.10 to be safe.

 # TODO: this is not yet enabled, pending testing

 #if ($driverBuild < 6476)

 boolProp useRenderTextures true

 #endif

 # avoid a race condition with color copies and ui

 boolProp presentWorkaround true

 if (match("${cardName}", "\*Radeon?VE\*") or match("${cardName}", "\*7?00\*") or match("${cardName}", "\*R100\*") or match("${cardName}", "\*IGP 3?0\*") or match("${cardName}", "\*9100 IGP\*"))

 log $logGroup $logLevelInfo "Forcing turbo rects off"

 setb supportsTurboRect false

 boolProp simpleTerrain true

 boolProp causticsEnabled false

 boolProp enumerateMultisampleLevels false

 setb forceLowSettings true

 if (match("${cardName}", "\*9100 IGP\*"))

 log $logGroup $logLevelInfo "Forcing low resolution"

 setb defaultLowResolution true

 endif

 if (match("${cardName}", "\*Radeon?VE\*") or $driverBuild < 6414)

 log $logGroup $logLevelInfo "Forcing swvp"

 boolProp forceSoftwareVP true

 endif

 endif

 if (match("${cardName}", "\*X1300\*") or match("${cardName}", "\*X300\*") or match("${cardName}", "\*XPRESS 200\*"))

 log $logGroup $logLevelInfo "Forcing medium settings or lower"

 boolProp useShaders false

 setb forceLowSettings true

 endif

 boolProp usePS30 false

 if ($maxVertexProgramVersionHWMajor < 2)

 # Radeon 8500 and greater have problems switching between FF and VS?

 log $logGroup $logLevelInfo "Forcing shaders of for ATI DX8"

 boolProp useShaders false

 endif

 if (match("${cardName}", "\*8500\*"))

 #avoid z fighting with light snow

 boolProp raiseLightSnowFF true

 setb defaultLowResolution true

 endif

 if (match("${cardName}", "\*9600 SE \*"))

 setb forceMediumMaterialDetail true

 setb forceMediumShadows true

 setb forceMediumObjectDetail true

 endif

 if (match("${cardName}", "\*Mobility\*"))

 setb isIntegratedChipset true

 endif

 # X800 based cards exhibit rendering corruption if turbo rect is enabled

 if (match("${cardName}", "\*X800\*") or match("${cardName}", "\*R420\*"))

 setb supportsTurboRect false

 log $logGroup $logLevelInfo "Forcing turbo rects off"

 endif

 if (match("${cardName}", "\*92?0\*"))

 log $logGroup $logLevelInfo "Forcing shadow detail on 9250 and 9200"

 setb forceMediumShadows true

 endif

####################### S3 ######################################################################

 elseif (match("${cardVendor}", "S3"))

 if (match("${cardName}", "\*GammaChrome\*") or match("${osVersion}","\*NT 6.0\*") )

 boolProp useShaders false

 log $logGroup $logLevelInfo "Forcing Gamma Chrome to use fixed function shaders"

 endif

 # EP6 CATlab reports gray screen during all cinematics

 if (match("${cardName}", "\*S27\*"))

 setOption SpecialEventCamera $Off

 endif

####################### INTEL ######################################################################

 elseif (match("${cardVendor}", "Intel"))

 if (match("${cardName}", "\*X3000\*"))

 boolProp disableVSyncSupport true # work around flickering UI

 else

#kiri remove auto default to low - Change for Intel (R) HD Graphics

# boolProp simpleTerrain true

# boolProp enumerateMultisampleLevels false

# boolProp disableVSyncSupport true # work around flickering UI

# boolProp useShaders false # (EP2 change) mostly for performance, but driver issues showed up in v14.14

 logSystemInfo "Kiri: Force Low settings because Intel removed"

 endif

 # the Intel minspec driver doesn't misreport available texture memory, so it's not

 # necessary to adjust the texture memory estimate it returns. This may also fix

 # a Windows "device failure" message that occurs sometimes on this device.

 boolProp disableTexMemEstimateAdjustment true

 # assuming Intel parts are UMA, drop the texture memory to 32 if not much system memory

 if ($memory <= $memoryLevelLow)

 if ($textureMemory > 32)

 seti textureMemory 32

 endif

 endif

 if (match("${cardName}", "\*845\*") or match("${cardName}", "\*865\*") or match("${cardName}", "\*830\*") or match("${cardName}", "\*855\*"))

 boolProp causticsEnabled false

 setb forceLowSettings true

 setb forceLowResolution true

 endif

 # the 915 sets the caps bit that indicates it can do texture projection,

 # but does it incorrectly in the pixel stage. Here we override the texture

 # projection cap we get from the device.

 if (match("${cardName}", "\*915\*"))

 boolProp disableTextureProjection true

 endif

 endif

 intProp deviceLocalVideoMemorySizeDefault ($textureMemory \* 1024 \* 1024)

####################### OTHER ######################################################################

else

 setb forceLowSettings true

 setb forceLowResolution true

 # force sw, swvp, and no aa

 boolProp forceSoftwareDevice true

 boolProp forceSoftwareVP true

 boolProp enumerateMultisampleLevels false

 boolProp simpleTerrain true

 boolProp causticsEnabled false

 intProp softwareDeviceLocalVideoMemorySize ($textureMemory \* 1024 \* 1024)

endif

# since we don't do bumpmapping on less ps2.0-capable hardware, eliminate tangents

# from the vertex data

# also, the presence of per-vertex tangents in the vertex data stream causes

# "exploded" polygons on the Radeon 9000, even if the data is ignored by the shader

if ($maxPixelProgramVersionMajor < 2)

 boolProp skipTangentsInVertexData true

endif

#############################################################################

#

# Print system info

#

if ($deviceNumber = 0)

logSystemInfo "=== Application info ==="

logSystemInfo "Name: ${appName}"

logSystemInfo "Version: ${version}"

logSystemInfo "Build: ${buildType}"

logSystemInfo "=== Machine info ==="

logSystemInfo "OS version: ${osVersion}"

logSystemInfo "CPU: ${cpuSpeed}Mhz, Name:${CPU}, FPU:${FPU}, MMX:${MMX}"

logSystemInfo "Memory: ${memory}MB"

logSystemInfo "Free memory: ${freeMemory}MB"

logSystemInfo "User: ${userName}"

logSystemInfo "Computer: ${computerName}"

logSystemInfo "=== Sound device info ==="

logSystemInfo "Name: ${soundCardName}"

logSystemInfo "Driver: ${soundDriverName}"

endif

logSystemInfo "=== Graphics device info ==="

logSystemInfo "Number: ${deviceNumber}"

logSystemInfo "Name (driver): ${cardNameText}"

if ($isCardFound)

logSystemInfo "Name (database): ${cardName}"

else

logSystemInfo "Name (database): ${cardName} <<NOT FOUND IN DATABASE!>>"

endif

if ($isIntegratedChipset)

logSystemInfo " (Integrated Chipset)"

endif

logSystemInfo "Vendor: ${cardVendor}"

logSystemInfo "Chipset: ${cardChipset}"

logSystemInfo "Driver: ${driverName}, Version: ${driverVersion}"

logSystemInfo "Driver version: ${driverBuild}"

logSystemInfo "Monitor: ${monitorName}"

logSystemInfo "Monitor aspect: ${monitorAspect}, ${monitorAspectString}"

logSystemInfo "Screen mode: ${screenWidth}x${screenHeight}x${screenBPP}BPP,${screenRefresh}Hz"

if ($textureMemorySizeOK)

logSystemInfo "Texture memory: ${textureMemory}MB"

else

logSystemInfo "Texture memory: ${textureMemory}MB <<OVERRIDE>>"

endif

logSystemInfo "HW T&L: Fixed function:${fixedFunctionHWTnL} Programmable:${maxVertexProgramVersionHWMajor}.${maxVertexProgramVersionHWMinor}"

logSystemInfo "Pixel program: ${maxPixelProgramVersionMajor}.${maxPixelProgramVersionMinor}"

logSystemInfo "Texture stages: ${textureStages}"

logSystemInfo "AppControlledAA: ${appControlledAA}"

#############################################################################

#

# UI Options

log $logGroup $logLevelDebug "Begin parsing option definitions"

option MaterialDetail

 setting $Low

 boolProp bumpMapping false

 intProp imageDataSizeReductionOnLoad 2

 setting $Medium

 boolProp bumpMapping false

 intProp imageDataSizeReductionOnLoad 1

 setting $High

 boolProp bumpMapping true

 intProp imageDataSizeReductionOnLoad 0

end

option ObjectDetail

 setting $Medium

 boolProp reduceBoneWeights true

 boolProp useLODs true

 intProp lodOverride 35

 setting $High

 boolProp reduceBoneWeights false

 boolProp useLODs false

 intProp lodOverride 0

end

option ObjectHiding

 setting $Off

 intProp renderInsideVisibleObjects 0

 setting $On

 intProp renderInsideVisibleObjects 1

end

option SnowOnGround

 setting $Off

 boolProp showSnowOnGround false

 setting $On

 boolProp showSnowOnGround true

end

option Shadows

 setting $Low

 boolProp simShadows false

 boolProp objectShadows false

 boolProp guob false

 boolProp heightMapShadows false

 setting $Medium

 boolProp simShadows false

 boolProp objectShadows false

 boolProp guob true

 boolProp heightMapShadows true

 setting $High

 boolProp simShadows true

 boolProp objectShadows true

 boolProp guob true

 boolProp heightMapShadows true

end

option OpaqueUI

 setting $Off

 boolProp renderOpaqueUI false

 setting $On

 boolProp renderOpaqueUI true

end

option Reflection

 setting $Off

 boolProp reflectionWithExtraViewer false

 boolProp nhoodWaterReflection false

 setting $On

 boolProp reflectionWithExtraViewer true

 boolProp nhoodWaterReflection true

end

option Ceiling

 setting $Off

 boolProp includeCeilings false

 setting $On

 boolProp includeCeilings true

end

option EffectsQuality

 setting $Low

 boolProp useEffects true

 floatProp particleDensity 1

 floatProp particleScale 1

 intProp maxParticlesTarget 2000

 intProp particleLODOffset 0 # subtracted from the zoom.

 intProp effectPriorityLevel 1

 boolProp enableOceanReflection false

 setting $Medium

 boolProp useEffects true

 floatProp particleDensity 1

 floatProp particleScale 1

 intProp maxParticlesTarget 5000

 intProp particleLODOffset 0 # subtracted from the zoom.

 intProp effectPriorityLevel 2

 boolProp enableOceanReflection false

 setting $High

 boolProp useEffects true

 floatProp particleDensity 1

 floatProp particleScale 1

 intProp maxParticlesTarget 10000

 intProp particleLODOffset 0 # subtracted from the zoom.

 intProp effectPriorityLevel 3

 boolProp enableOceanReflection true

end

option LightingQuality

 setting $Low

 boolProp lightingEnabled true

 boolProp portalLighting false

 boolProp floorAndWallNormalMapping false

 boolProp specHighlights false

 setting $Medium

 boolProp lightingEnabled true

 boolProp portalLighting true

 boolProp floorAndWallNormalMapping false

 boolProp specHighlights true

 setting $High

 boolProp lightingEnabled true

 boolProp portalLighting true

 boolProp floorAndWallNormalMapping true

 boolProp specHighlights true

end

option SoundQuality

 setting $Low

 intProp AudioPerformance 0

 setting $Medium

 intProp AudioPerformance 1

 setting $High

 intProp AudioPerformance 2

end

#

# Options without ui access.

#

option DirtyRect

 setting $High

 # no dirty rects

 intProp dynamicRenderStrategy 0

 setting $Medium

 # 4 buffer mode

 intProp dynamicRenderStrategy 2

 setting $Low

 # 2 buffer mode

 intProp dynamicRenderStrategy 1

end

option FullscreenFadeEffect

 setting $Off

 boolProp enableSnapshot false

 setting $On

 boolProp enableSnapshot true

end

option Turbo

 setting $High

 boolProp useTurboRect true

 setting $Medium

 boolProp useTurboRect true

 setting $Low

 boolProp useTurboRect false

end

option SimulatorControls

 setting $High

 intProp maxNumOfVisitingSims 8

 setting $Medium

 intProp maxNumOfVisitingSims 6

 setting $Low

 intProp maxNumOfVisitingSims 2

end

option LightingOptimizations

 setting $High

 boolProp optimizedDiffusion true

 boolProp incrementalLighting true

 boolProp lerpLights true

 boolProp useDirtyTiles true

end

option AnimationSamplingLevel

 setting $Low

 # sample nearest animation frame

 boolProp animationFrameSampling true

 setting $Medium

 # sample nearest two animation frames and slerp

 boolProp animationFrameSampling false

end

option LivePIP

 setting $Off

 boolProp livePIP false

 boolProp livePIPDefault false

 setting $On

 boolProp livePIPDefault true

 # do not override user's choice if the setting is ON

end

option SpecialEventCamera

 setting $Off

 boolProp chooseCameraSpecialEventEnabled false

 boolProp CameraSpecialEventEnabled false

 setting $On

 boolProp chooseCameraSpecialEventEnabled true

 # do not override user's choice if the setting is ON

end

option ScreenModeResolution

 setting $Low

 uintProp maxResWidth 800

 uintProp maxResHeight 600

 uintProp defaultResWidth 800

 uintProp defaultResHeight 600

 setting $MediumButDefaultLow

 uintProp maxResWidth 1280

 uintProp maxResHeight 1024

 uintProp defaultResWidth 800

 uintProp defaultResHeight 600

 setting $Medium

 uintProp maxResWidth 1280

 uintProp maxResHeight 1024

 uintProp defaultResWidth 1024

 uintProp defaultResHeight 768

 setting $High

 uintProp maxResWidth 1600

 uintProp maxResHeight 1200

 uintProp defaultResWidth 1024

 uintProp defaultResHeight 768

end

option SubjectTracking

 setting $Low

 floatProp centerTrackingDeadZoneMagnitude 70

 setting $Medium

 floatProp centerTrackingDeadZoneMagnitude 30

 setting $High

 floatProp centerTrackingDeadZoneMagnitude 30

end

option EnableLotImpostersInLot

 setting $On

 boolProp enableLotImpostersInLot true

 setting $Off

 boolProp enableLotImpostersInLot false

end

option EnableNeighborhoodOccupantsInLot

 setting $On

 boolProp enableNeighborhoodOccupantsInLot true

 setting $Off

 boolProp enableNeighborhoodOccupantsInLot false

end

option LotSkirtSizeIncrease

 setting 0

 uintProp lotSkirtSizeIncrease 5

 setting 1

 uintProp lotSkirtSizeIncrease 9

 setting 2

 uintProp lotSkirtSizeIncrease 18

 setting 3

 uintProp lotSkirtSizeIncrease 36

end

log $logGroup $logLevelDebug "Finished parsing option definitions"

#############################################################################

#

# Hardware Rendering

#

# This cannot be defined as a local variable, the if clause is still executed when

# sw render path is taken and an exception generated that fails the parsing. Be

# careful with the use of local variables.

setb usingHighDetail false

if (not $useSoftwareRasterizer)

 # set a base level of options based on the card rating

 if (($memory >= $memoryLevelHigh) and ($cpuSpeed >= $cpuLevelHigh) and ($maxVertexProgramVersionHWMajor >= 1) and not $forceLowSettings and not $isIntegratedChipset)

 log $logGroup $logLevelInfo "Selecting High base level"

 logSystemInfo "Kiri: Selecting High base level"

 setOption OpaqueUI $Off

 setOption LivePIP $On

 setOption Reflection $On

 setOption Ceiling $On

 setOption EffectsQuality $High

 setOption LightingQuality $High

 setOption Shadows $High

 setOption MaterialDetail $High

 setOption ObjectDetail $High

 setOption ObjectHiding $Off

 setOption SubjectTracking $High

 setOption EnableLotImpostersInLot $On

 setOption EnableNeighborhoodOccupantsInLot $On

 setOption LotSkirtSizeIncrease 1

 setb usingHighDetail true #use this to test if current level is set to high

 elseif (($memory >= $memoryLevelMedium) and ($cpuSpeed >= $cpuLevelMedium) and ($maxVertexProgramVersionHWMajor >= 1) and not $forceLowSettings)

 log $logGroup $logLevelInfo "Selecting Medium base level"

 logSystemInfo "Kiri: Selecting Medium base level"

 setOption OpaqueUI $Off

 setOption LivePIP $On

 setOption Reflection $Off

 setOption Ceiling $Off

 setOption EffectsQuality $Medium

 setOption LightingQuality $Medium

 setOption Shadows $Medium

 setOption MaterialDetail $Medium

 setOption ObjectDetail $High

 setOption ObjectHiding $On

 setOption SubjectTracking $Medium

 setOption EnableLotImpostersInLot $On

 setOption EnableNeighborhoodOccupantsInLot $On

 setOption LotSkirtSizeIncrease 0

 else

 log $logGroup $logLevelInfo "Selecting Low base level"

 logSystemInfo "Kiri: Selecting Low base level"

 # NOTE: don't set forceLowSettings here, because this will force low simulator settings

 # if all you have is a low end video card

 setOption OpaqueUI $Off

 setOption LivePIP $Off

 setOption Reflection $Off

 setOption Ceiling $Off

 setOption EffectsQuality $Low

 setOption LightingQuality $Low

 setOption Shadows $Low

 setOption MaterialDetail $Low

 setOption ObjectDetail $Medium

 setOption ObjectHiding $On

 setOption SubjectTracking $Low

 setOption EnableLotImpostersInLot $Off

 setOption EnableNeighborhoodOccupantsInLot $Off

 setOption LotSkirtSizeIncrease 0

 endif

 setOption LightingOptimizations $High

 setOption SnowOnGround $On

 # adjust simulator based on cpu speed

 # adjust the sound quality based on cpu speed (this may be overridden below for low-end systems)

 # adjust animation sampling based on cpu speed

 if ($cpuSpeed >= $cpuLevelHigh and not $forceLowSettings)

 setOption SimulatorControls $High

 setOption SoundQuality $High

 setOption AnimationSamplingLevel $Medium

 elseif ($cpuSpeed >= $cpuLevelMedium and not $forceLowSettings)

 setOption SimulatorControls $Medium

 setOption SoundQuality $Medium

 setOption AnimationSamplingLevel $Medium

 else

 setOption SimulatorControls $Low

 setOption SoundQuality $Low

 setOption AnimationSamplingLevel $Low

 endif

 # turn off fullscreen fade

 setOption FullscreenFadeEffect $Off

 if ($textureMemory >= 128 and not $isIntegratedChipset)

 setOption FullscreenFadeEffect $On

 endif

 # screen res defaults

 if (($maxVertexProgramVersionHWMajor >= 1) and ($textureMemory >= 128) and not $forceLowResolution)

 setOption ScreenModeResolution $High

 elseif ($textureMemory >= 64 and not $forceLowResolution)

 if (($memory <= $memoryLevelLow) and ($cpuSpeed <= $cpuLevelLow))

 setOption ScreenModeResolution $MediumButDefaultLow

 else

 setOption ScreenModeResolution $Medium

 endif

 else

 setOption ScreenModeResolution $Low

 endif

 if ($defaultLowResolution)

 setOption ScreenModeResolution $MediumButDefaultLow

 endif

 # special event cameras for cinematics

 if (($maxPixelProgramVersionMajor >= 1) and ($supportsSpecialEventCamera))

 setOption SpecialEventCamera $On

 else

 setOption SpecialEventCamera $Off

 endif

 # set dirty rect mode

 if ($supportsDirtyRect)

 setOption DirtyRect $Medium

 else

 setOption DirtyRect $High

 endif

 # set turbo mode

 if ($supportsTurboRect)

 setOption Turbo $Medium

 else

 setOption Turbo $Low

 endif

 if ($forceMediumMaterialDetail and ($memory >= $memoryLevelHigh) and ($cpuSpeed >= $cpuLevelHigh) and not $forceLowSettings)

 log $logGroup $logLevelWarning "Setting Material Detail"

 setOption MaterialDetail $Medium

 endif

 if ($usingHighDetail and not $forceLowSettings)

 if ($forceMediumShadows)

 setOption Shadows $Medium

 endif

 if ($forceMediumObjectDetail)

 setOption ObjectDetail $Medium

 endif

 endif

endif

#############################################################################

#

# Software rendering

#

# set the options for the software rasterizer

if ($useSoftwareRasterizer)

 setOption LightingOptimizations $High

 setOption DirtyRect $Medium

 setOption Turbo $Medium

 setOption OpaqueUI $Off

 setOption LivePIP $Off

 setOption SpecialEventCamera $Off

 setOption Reflection $Off

 setOption Ceiling $Off

 setOption EffectsQuality $Low

 setOption LightingQuality $Low

 setOption Shadows $Low

 setOption MaterialDetail $Low

 setOption ObjectDetail $Medium

 setOption ObjectHiding $On

 # adjust simulator based on cpu speed

 if ($cpuSpeed >= $cpuLevelHigh)

 setOption SimulatorControls $Medium # one lower than hw

 else

 setOption SimulatorControls $Low

 endif

 setOption SoundQuality $Low

 setOption FullscreenFadeEffect $Off

 setOption AnimationSamplingLevel $Low

 setOption ScreenModeResolution $Low

 setOption SubjectTracking $Low

 setOption EnableLotImpostersInLot $Off

 setOption EnableNeighborhoodOccupantsInLot $Off

 setOption LotSkirtSizeIncrease 0

 setOption SnowOnGround $On

endif

#############################################################################

# save the selected configuration for this device.

boolProp activeDeviceUseSoftwareRasterizer $useSoftwareRasterizer

#############################################################################

# tell the caller that we completely finished parsing the script

intProp configParserErrorCode 0

uintProp hwMajorPSVersion $maxPixelProgramVersionMajor

log $logGroup $logLevelInfo "Finished Config File"